



Technology Consulting Company
Research, Development &
Global Standard

Integrating Video Processing Hardware into GStreamer

- a case study of the Renesas R-CarE1 platform

2012年12月7日

松原 克弥
小林 和徳
株式会社イーゲル

クロスプラットフォーム対応マルチメディアフレームワーク

- Linux系プラットフォームでは、デファクトスタンダード
 - GStreamer for Android
 - Tizen
- プラグインと呼ばれるコンポーネントを接続したパイプラインを構成することで、マルチメディア処理を記述
- 200以上のプラグインが存在
 - core, base, good, bad, uglyに分類
 - gst-openmax, gst-ffmpeg等の3rd partyプラグインも存在
- gst-launch: 容易にパイプライン構築が可能なテストツール

Plugins (Elements)

機能に応じて3種類のElementに分類

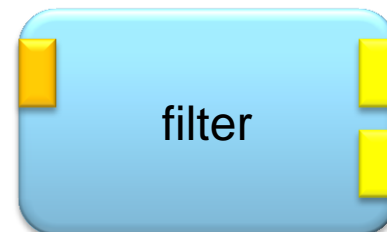
■ Source element

- filesrc : ファイルからデータ入力
- fakesrc : 空データを生成
- videotestsrc : テスト用ビデオストリームを生成



■ Filter or filter-like element

- ffdec_h264 : H.264ビデオのデコード
- videocrop : 画像のクロップ
- qtdemux : mp4, mov, 3gp等のコンテナをdemuxして、Audio/Videoストリームを抽出



■ Sink element

- fbdevsink, xvimagesink, dfbvideosink : 画面出力
- filesink : ファイル出力



GStreamer Overview (contd.)

■ アプリケーションAPIとプラグインAPIを提供

- GStreamer Application Development Manual

<http://gstreamer.freedesktop.org/data/doc/gstreamer/0.10.36/manual/html/>

- GStreamer Plugin Write's Guide

<http://gstreamer.freedesktop.org/data/doc/gstreamer/0.10.36/pwg/html/>

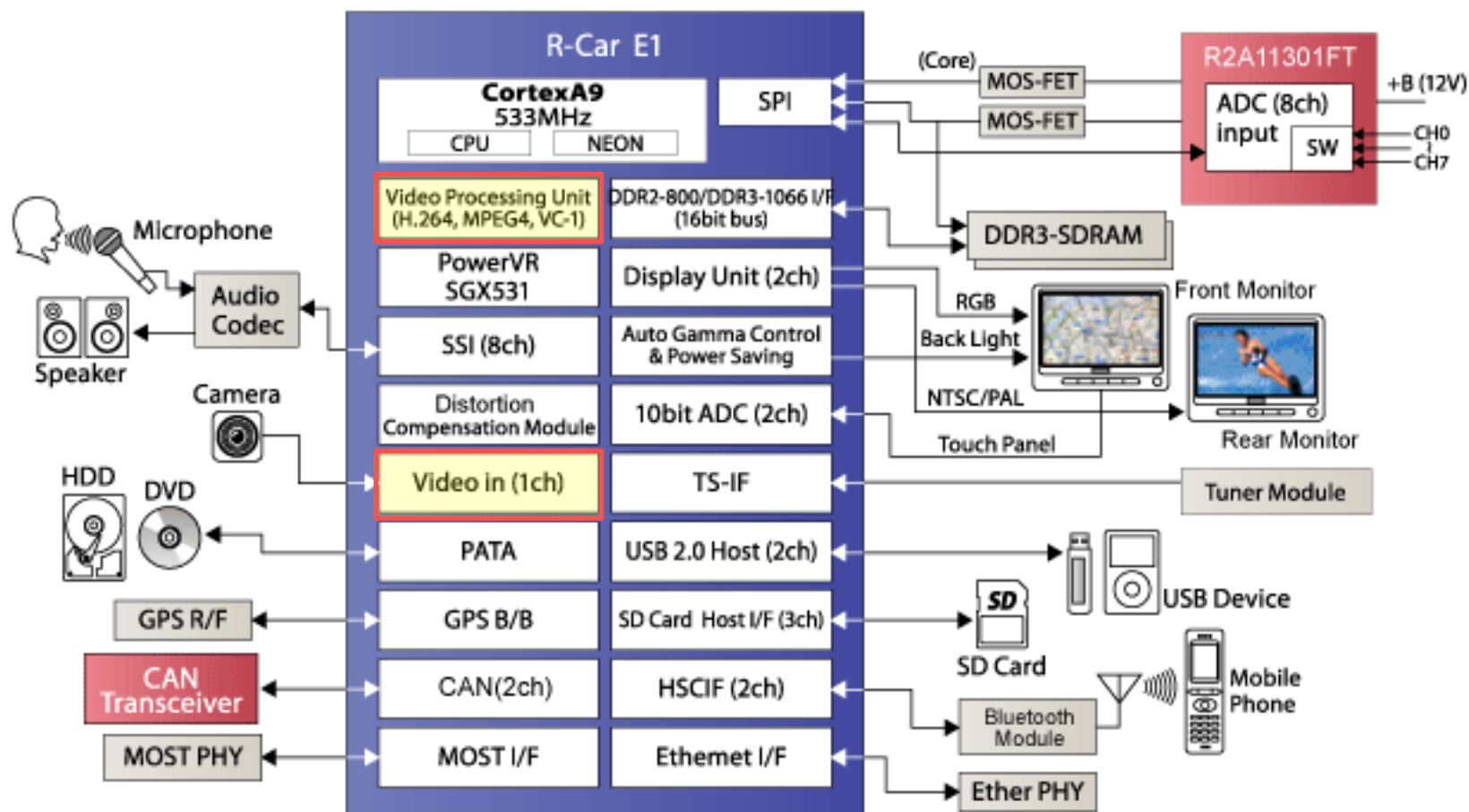
■ 2系統のバージョンが現存

0.10／現在も多く使われている旧stableバージョン、最終は0.10.36

1.0 (旧0.11)／以前はunstableと呼ばれていた開発バージョンで2012年9月に1.0としてstable化、最新は1.0.3

本発表はバージョン0.10.36を対象

Renesas R-Car E1



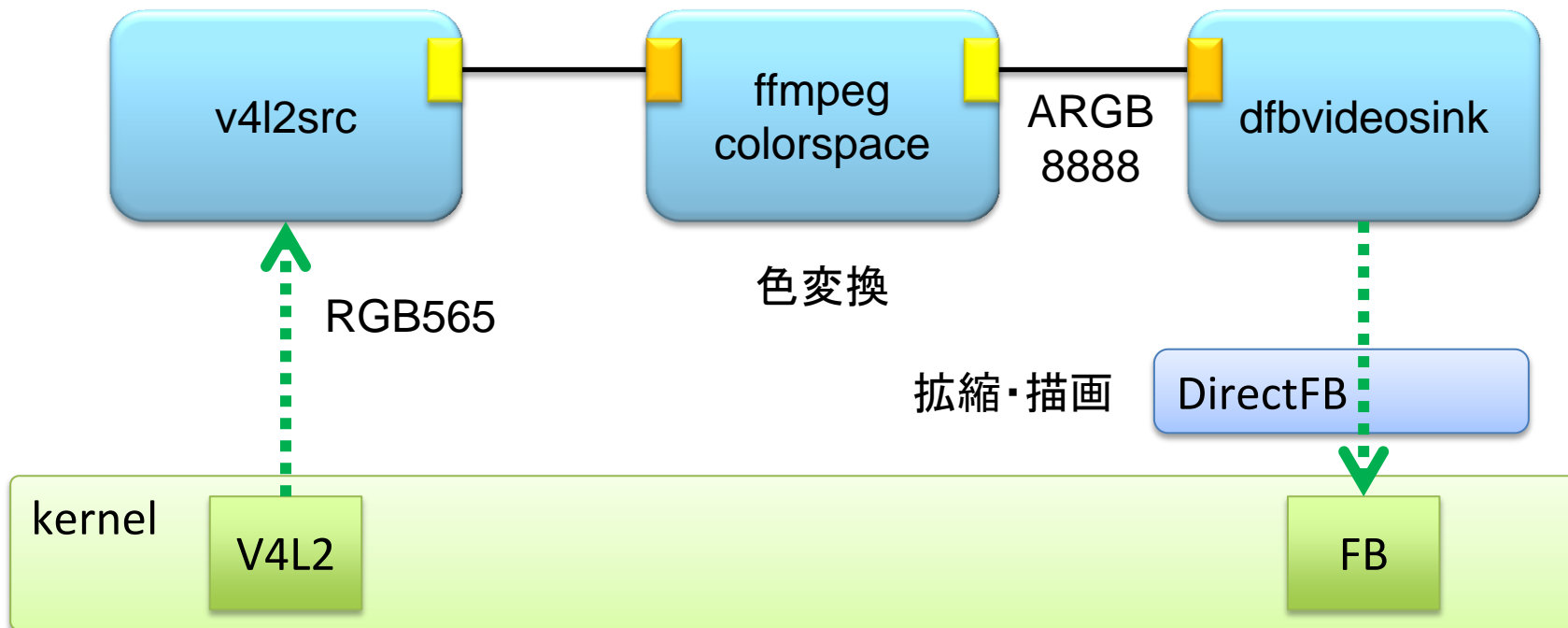
http://japan.renesas.com/applications/automotive/cis/cis_highend/rcar_e1/index.jsp より引用

VIDEO CAPTURE (VIDEO IN)

Pipeline for Video Capturing

v4l2src ! ffmpegcolourspace ! ¥

video/x-raw-rgb, bpp=32, depth=32, ... ! dfbvideosink



解析:なぜ遅いのか

1. v4l2src :
V4L2バッファからGSTバッファへmemcpy
– V4L2バッファをすばやく解放するため
2. ffmpegcolospace :
RGB565からARGB8888への色変換がS/W処理
3. dfbvideosink :
画像拡張とフレームバッファへの描画がS/W処理
4. V4L2 VINドライバ :
バッファ数不足で低速な(15fps)キャプチャモードを選択
– バッファ数 ≥ 4 で高速(30fps)キャプチャモードが動作
– v4l2srcが確保するデフォルト・バッファ数は2

v4l2srcのチューニング

1. memcpyの抑制

4. V4L2バッファ数の変更

Properties in Plugin

■ プラグインのインスタンス毎に指定できるパラメータ

“gst-inspect v4l2src”の結果抜粋

Element Properties:

...

queue-size	: Number of buffers to be enqueued in the driver in streaming mode flags: readable, writable Unsigned Integer. Range: 1 - 16 Default: 2 Current: 2
always-copy	: If the buffer will or not be used directly from mmap flags: readable, writable Boolean. Default: true Current: true

dfbvideosink: 画像処理H/Wによる描画

2. 色変換処理 3. 拡張を伴った描画

主な内蔵
周辺機能

- ビデオ入力インタフェース×1チャンネル
- VPU5HD2 (H.264/AVC、MPEG-4、VC-1)
- ビデオ画像処理機能 (色変換、画像拡大・縮小、フィルタ処理)
- 歪み補正モジュール
- SDホストインタフェース ×3チャンネル
- マルチメディアカードインタフェース
- 各種サウンドインタフェースI/O×8チャンネル
- メディアローカルバス(MLB)インタフェース×1チャンネル
(MediaLB Ver2.0に準拠、512fs(max)のデータ転送が可能)
- USB 2.0 ホストインタフェース ×2ポート
- GPS ベースバンド処理モジュール
- TS インタフェース
- CD-ROMデコーダ

これ使います

http://japan.renesas.com/applications/automotive/cis/cis_highend/rcar_e1/index.jsp より引用

- UIO(Userspace IO)でユーザランドへIPを解放
- libshvioとしてVIO制御ミドルウェアを実装

H/W処理に適したメモリの使用と H/W IP間メモリ共有

要件

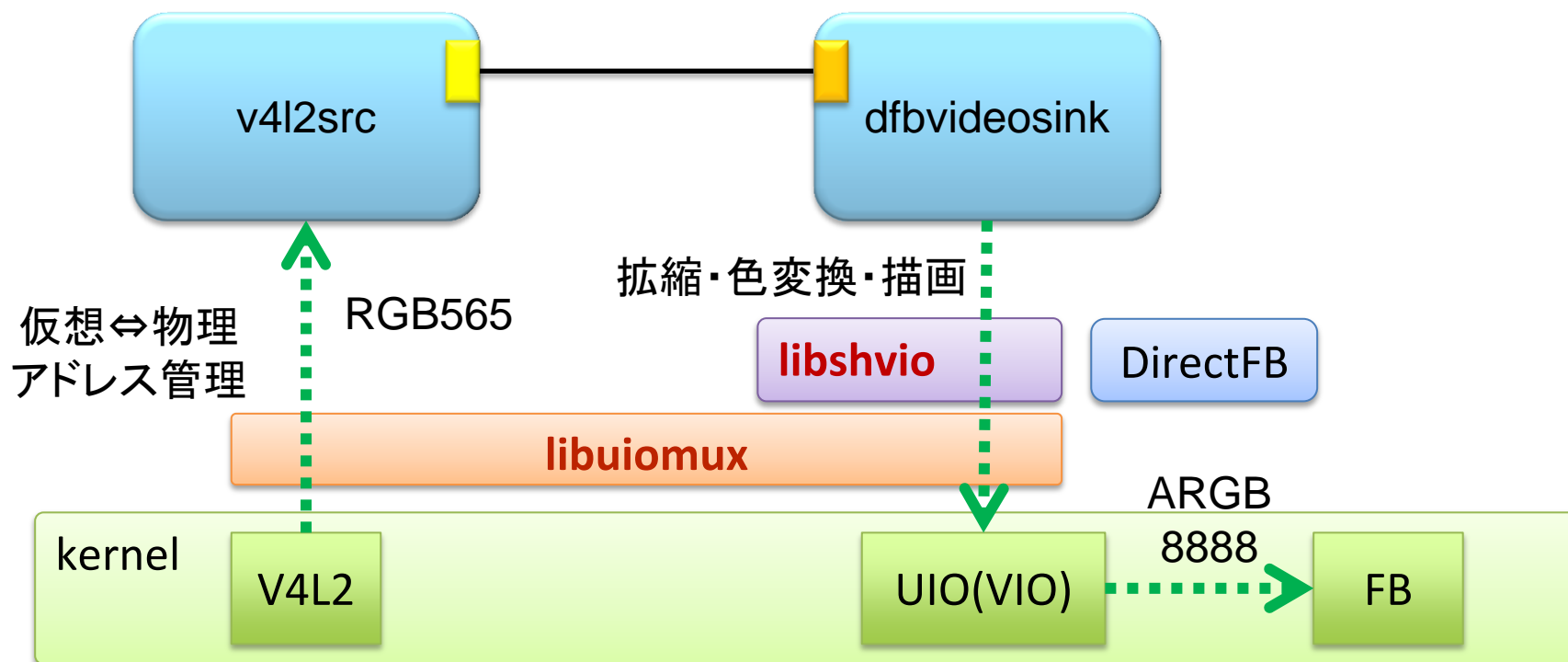
- VINもVIOも、物理連続メモリが必要
- バッファの仮想⇔物理アドレス変換が必要
 - VINはV4L2カーネルドライバで、ユーザランドにはバッファの仮想アドレスのみを扱う
 - VIOはlibshvioユーザランド・ドライバなので、ユーザランドで物理アドレスが必要

方策

- 連続物理メモリの確保は**UIO**で実現
- 仮想⇔物理アドレス変換は**libuio mux**が管理
- V4L2 VINドライバへの上記メモリの指定は**V4L2_MEMORY_USERPTR**を使用

Pipeline for Video Capturing (revised)

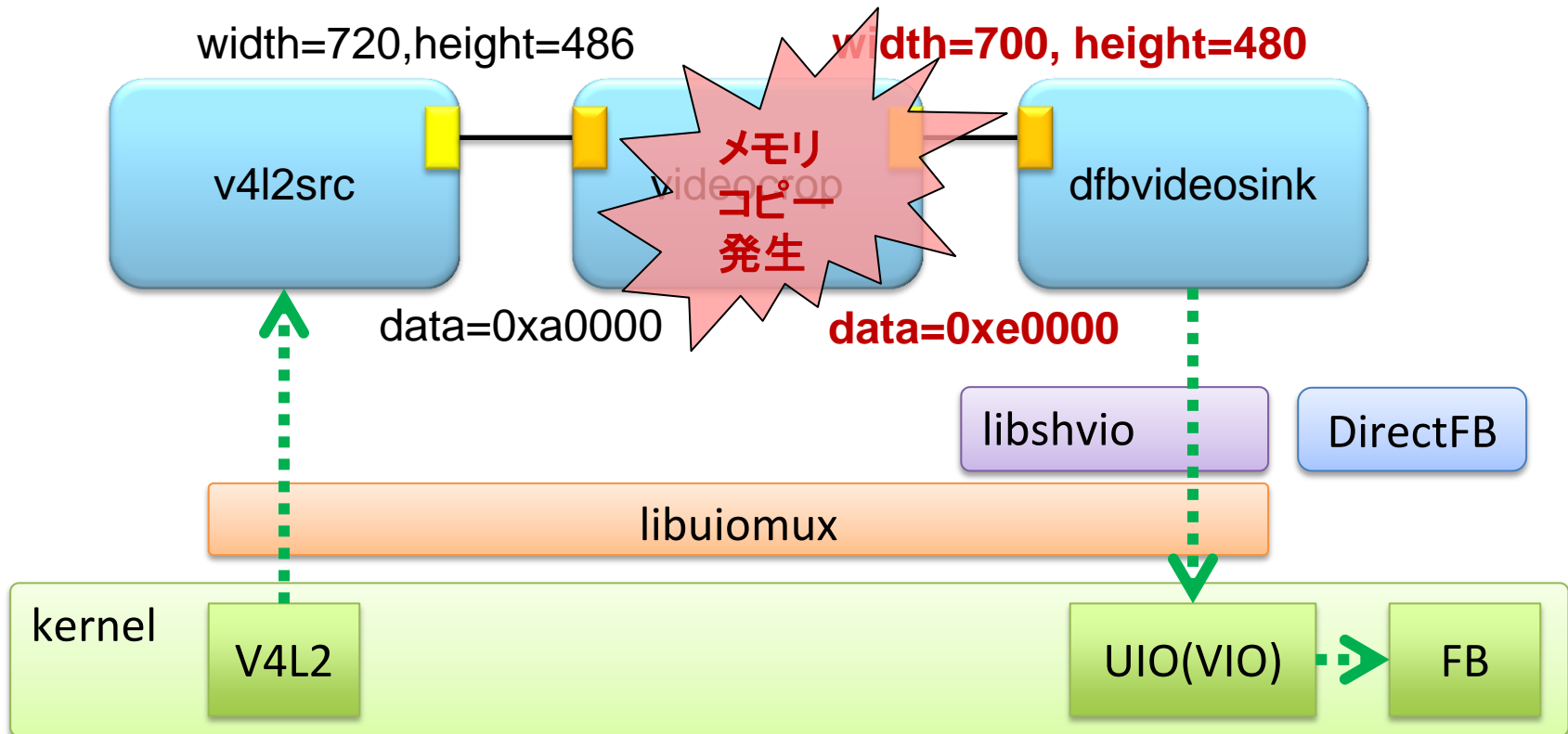
v4l2src queue-size=5 always-copy=false ! dfbvideosink



Pipeline for Video Capturing with Image Cropping

v4l2src queue-size=5 always-copy=false ! ¥

videocrop top=6 left=10 right=10 ! dfbvideosink



Cropping without memcpy

- 描画H/Wの多くは、pitch/strideの指定に対応



Capabilities (caps)

Elementの入出力(pads)に付加されたデータ

- 画像サイズ、色フォーマット、コーデック、フォーマット等のデータに対するメタデータ保持に利用
- decodebin, playbin等のプラグインで適切なelementの自動検索に利用されたり、接続されたelement間の互換性確認に使われる。
- 任意のパラメータを追加可能、ただし、追加されたパラメータに対応しないelementは無視する。

memcpyによるクロップ処理の代わりに
capsにrowstride=オリジナルwidth値を設定
→sinkでstrideを考慮した画像読み込み & 描画

前後のElementへ問い合わせを行う仕組み

- 隣のElementだけでなく、端まで順次問い合わせ
- 定義済問い合わせタイプだけでなく、任意の問い合わせを追加可能
- 返答はTRUE or FALSE

```
typedef enum {  
    GST_QUERY_NONE = 0,  
    GST_QUERY_POSITION,  
    GST_QUERY_DURATION,  
    GST_QUERY_LATENCY,  
    GST_QUERY_JITTER,  
    GST_QUERY_RATE,  
    GST_QUERY_SEEKING,  
    GST_QUERY_SEGMENT,  
    GST_QUERY_CONVERT,  
    GST_QUERY_FORMATS,  
    GST_QUERY_BUFFERING,  
    GST_QUERY_CUSTOM,  
    GST_QUERY_URI  
} QueryType;
```

後段(sink)がcaps"rowstride"に対応している
か問い合わせ:

if (対応している)

capsにrowstide=オリジナルwidth値を設定

else

従来のmemcpyによるクロップ処理

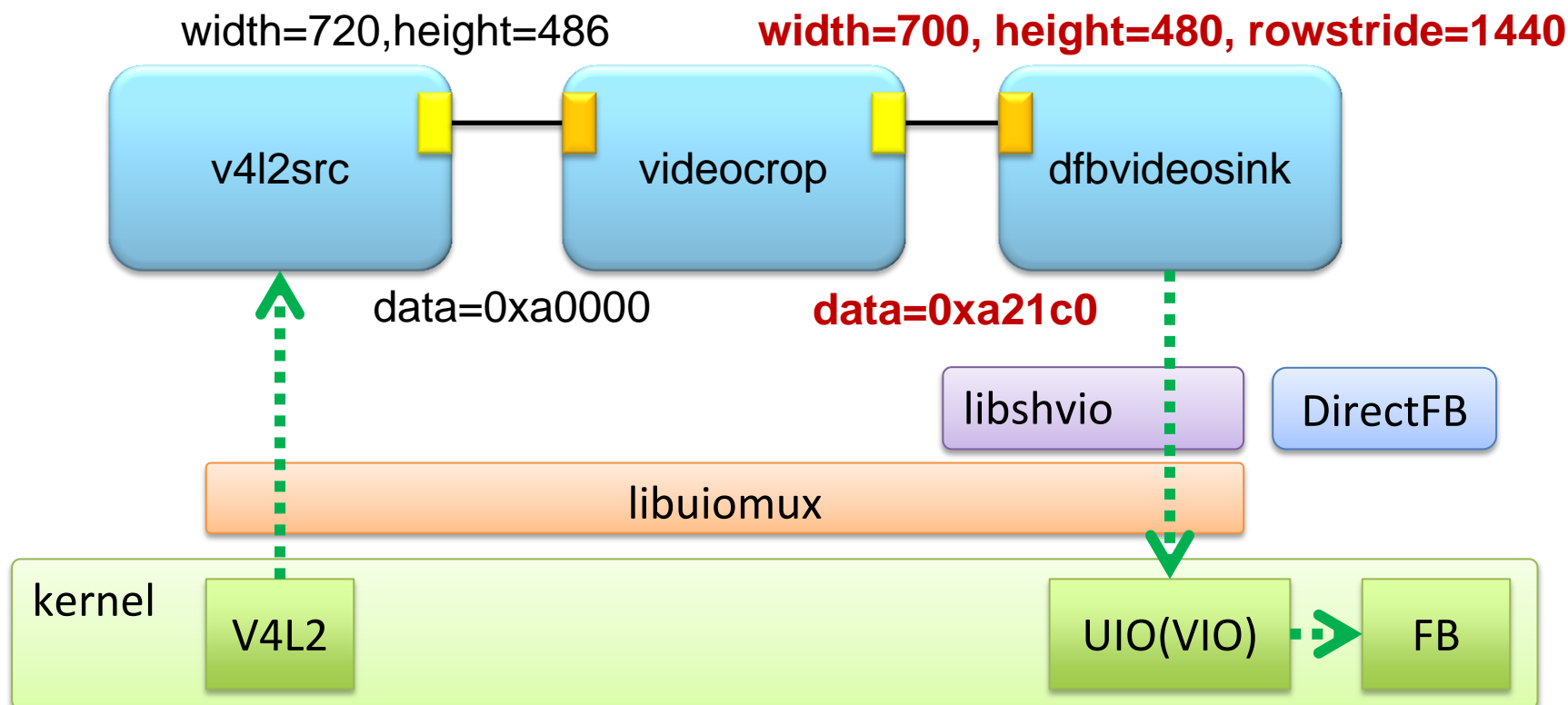
親バッファの複製、ただし、データのメモリ領域は親と共有

- 入力バッファの一部をSub-bufferとして出力することで、メモリコピーを抑制できる。
- 参照カウンタにより、Sub-bufferへの参照がなくなるまで、親バッファも(誤って)解放されない。

先頭アドレスを「top行数 + rightピクセル」分ずらしたSub-bufferを作成して、後段へ出力

Pipeline for Video Capturing with Image Cropping

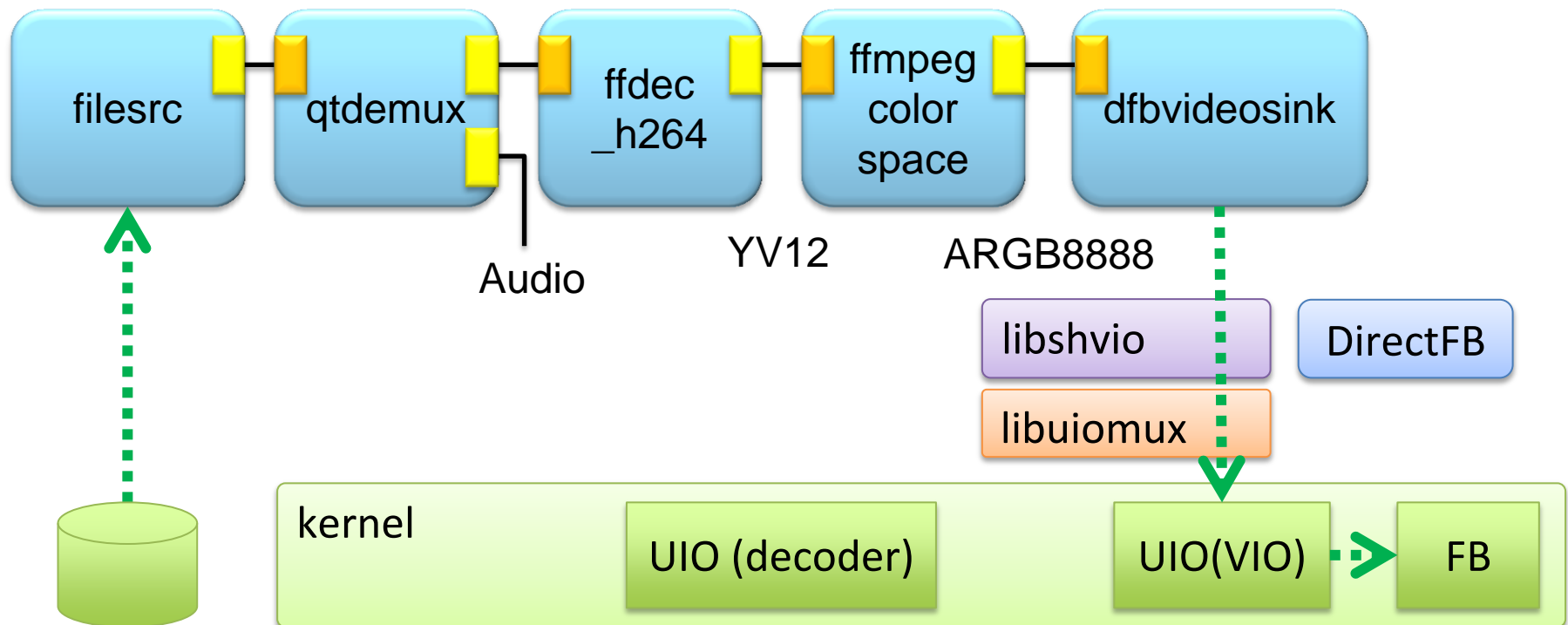
v4l2src queue-size=5 always-copy=false ! ¥
videocrop top=6 left=10 right=10 ! dfbvideosink



VIDEO PLAYING (DECODING)

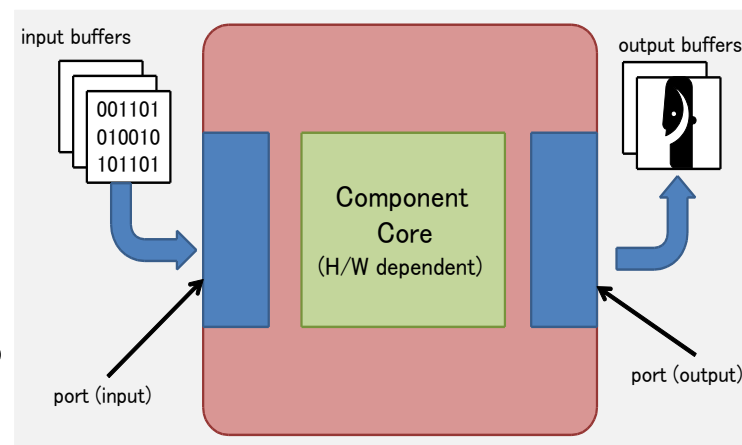
Pipeline for H.264 Video Playing

```
filesrc location=video.mp4 ! qtdemux name=dmx ¥  
dmx.video_00 ! ffdec_h264 ! ffmpegcolorspace ! ¥  
video/x-raw-rgb, bpp=32, .depth=32, .. ! dfbvideosink
```



OpenMAX IL (OMXIL)

- マルチメディア機能のためのAPI仕様
<http://www.khronos.org/openmax/>
- 特に、コーデック(デコーダ・エンコーダ)エンジンのStandard APIとして採用されていることが多い。
- AndroidにおけるコーデックエンジンI/FもOMXILを採用
- 非常にシンプルで緩いAPI仕様
 - GetParameter(), SetParameter()
 - UseBuffer(), AllocateBuffer()
 - FillThisBuffer(), EmptyThisBuffer()
- チップベンダorボードメーカーから配布されていることが多い。



GstreamerからOMXILコンポーネントを使うためのプラグイン

■ Filter Elements (decoder, encoder)

- H.263, H.264, MPEG4, WMV(VC-1)
- AAC, ADPCM, AMRNB, AMRWB, Vorbis, MP2, MP3, G711, G729
- JPEG, Volume

■ Source Element, Sink Element

■ OMXIL 1.1.1仕様に沿ったOMXIL Client実装

OMXIL仕様外の部分で
コンポーネント毎の合わせこみが必要！
cf. quirks in Android Stagefright



Integrating a Vender's OMXIL

■ データを入力仕様に合わせて加工

例: H.264デコード

- qtdemuxの出力仕様=フレーム単位、SPS/PPSはcapsを介して出力
- ffdec_h264の入力データ仕様=任意長のデータ入力可、SPS/PPSはcapsから入力
- omx_h264dec + REL OMXILの入力データ仕様=NAL単位、SPS/PPSもデータとして入力

■ バッファ確保方法を選択

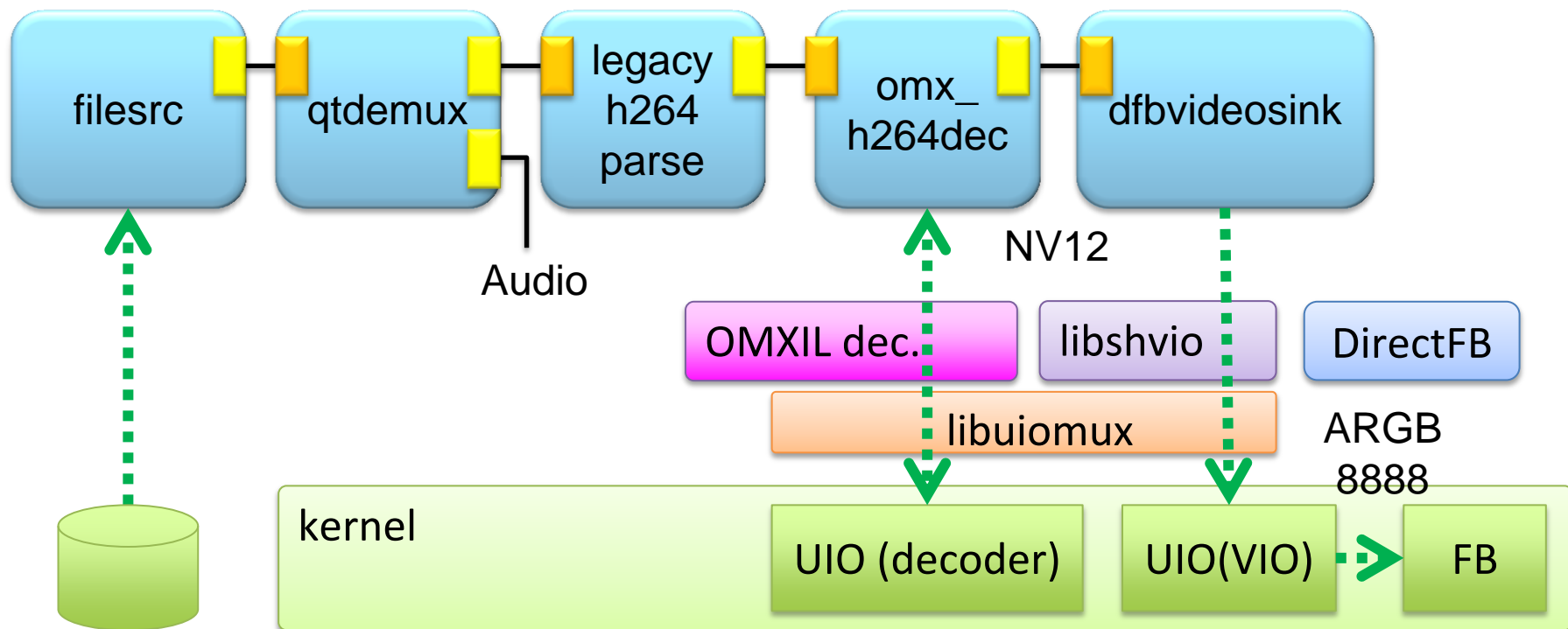
- UseBuffer() or AllocateBuffer()

Integrating a Vender's OMXIL (contd.)

- カスタムパラメータの設定(任意)
 - ベンダー独自モードやAPI
- シークやフラッシュ時の処理フローの差異
 - 例:フラッシュ実行前にPAUSEステートに移行すること
 - 例:シーク時はフラッシュを実行すること
- 出力仕様に合わせた調整
 - 画像幅のアライメント→caps:rowstrideへ設定
 - Tile-linear(T/L)アドレッシング→アドレッシングの有効／無効、タイル設定をcapsへ設定
 - インタレース画像出力: 合成 or Top-Bottom or Top or Bottom

Pipeline for H.264 Video Playing (revised)

```
filesrc location=video.mp4 ! qtdemux name=dmx ¥  
dmx.video_00 ! legacyh264parse output-format=1 ¥  
split-packetized=true ! omx_h264dec ! dfbvideosink
```



- ビデオH/Wインテグレーションは、以下の最適化が重要
 - H/W IPに適したメモリ(連続物理メモリ, 仮想⇔物理アドレス管理)の使用
 - (CPU)メモリコピーの削減
- 成果はgithubで公開
 - Patches for GStreamer plugins, libraries and kernel
<https://github.com/matsu/>
 - Install Guide for Renesas R-CarE1 Silverstone
<https://github.com/matsu/gst-openmax/wiki/Quick-Install-Guide-for-Renesas-R-CarE1-Silverstone>
- 今後の検討課題
 1. CMA, dma_buf, IOMMUの活用
 2. GStreamer 1.0への移行

(「ABS/ELC 2012 会議参加報告」 in Jamboree #40 より)

GStreamer 1.0: No Longer Compromise Flexibility ...



- By Edward Hervey, Collabora
- GStreamer 1.0で改善したメモリ管理に関する話
- 現バージョン 0.10 におけるメモリ管理
 - 1バッファ=1ポインタ
 - 非連続メモリが扱えない
 - CPUがアクセスできるメモリのみ対応
 - ハードウェアアクセラレーションのみが使うメモリ(アドレス)を扱えない
 - コンテンツはcapsで定義
 - 異なるレイアウト(たとえば、stride)を扱うために、カスタムcapsを定義
 - 固定されたGstBufferフィールド
 - メタデータを扱えない

(「ABS/ELC 2012 会議参加報告」 in Jamboree #40 より)

GStreamer 1.0: No Longer Compromise Flexibility ... (contd.)



- $\text{GstBuffer} = \text{GstMemory} \times n + \text{GstMeta} \times n$
- GstMemory: メモリの抽象化
 - フラグ、参照カウント、サイズ、アライメント、オフセットを管理
 - `map()`, `unmap()`によるアクセス
 - GstAllocator: `alloc`, `free`, `copy`, `share`, `map`, `unmap`を制御⇒ 様々なメモリを扱うことが可能
- GstMeta: バッファに関連するメタデータの管理
 - 様々な特性を定義可能
 - A/RGB, Y/Cb/Cr各プレーンやストライド等のビデオ情報
 - `crop`, `pan`などの処理情報
 - 各エレメントは必要なメタデータのみを参照